DOCUMENT RESUME

ED 057 558 EM 009 391

TITLE	Project Solo; Newsletter Number Ten.
INSTITUTION	Pittsburgh Univ., Pa. Dept. of Computer Science.
SPONS AGENCY	National Science Foundation, Washington, D.C.
PUB DATE	1 Feb 71
NOTE	25p.; See also ED 053 566
EDRS PRICE	MF-$0.65 HC-$3.29
DESCRIPTORS	*Computer Assisted Instruction; *Computer Programs; Mathematics Instruction; Simulation; *Trigonometry
IDENTIFIERS	*Project Solo

ABSTRACT

Four trigonometry modules of the Project Solo computer-assisted instruction series are presented. The modules deal with circular functions, trigonometry functions and Tchebychev polynomials, and inverse circular functions. A fourth module, VORTAC, allows students to try a two-aircraft navigation simulation. (JY)

ERIC

ED057558

PROJECT SOLO

AN EXPERIMENT IN REGIONAL COMPUTING FOR SECONDARY SCHOOL SYSTEMS*

University of Pittsburgh • Department of Computer Science •

U.S. DEPARTMENT OF HEALTH, EDUCATION & WELFARE
OFFICE OF EDUCATION
THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL OFFICE OF EDUCATION POSITION OR POLICY.

Newsletter No. 10 February 1, 1971

Category III Trigonometry Modules

We are enclosing four trigonometry modules. The module "CIRCULAR FUNCTIONS" was written by Mr. Robert Gillespie of Allderdice. Its place in mathematics is described in the overall plan he and his group prepared last summer. The "12th grade mathematics" topics given in Newsletter No. 7 came out of that plan.

As a result of mathematical questions raised by "CIRCULAR FUNCTIONS", two other modules were prepared called "TRIGONOMETRY FUNCTIONS AND TCHEBYCHEV POLYNOMIALS" and "INVERSE CIRCULAR FUNCTIONS". The module "VORTAC" is for the more ambitious student. Students who want to try a two aircraft simulation of navigating by VORTAC should type:

```
>RUN 166FW /WGAME/
```

Call For Student Programs

We would like to circulate student programs of all levels through the newsletter. This is not a request for "best programs", but rather a mechanism through which teachers can give recognition to experienced as well as neophyte student-users of computing. Please include a listing, sample run, and appropriate documentation, including teacher's name, subject, student's name, age, and grade.

Exchange Programs

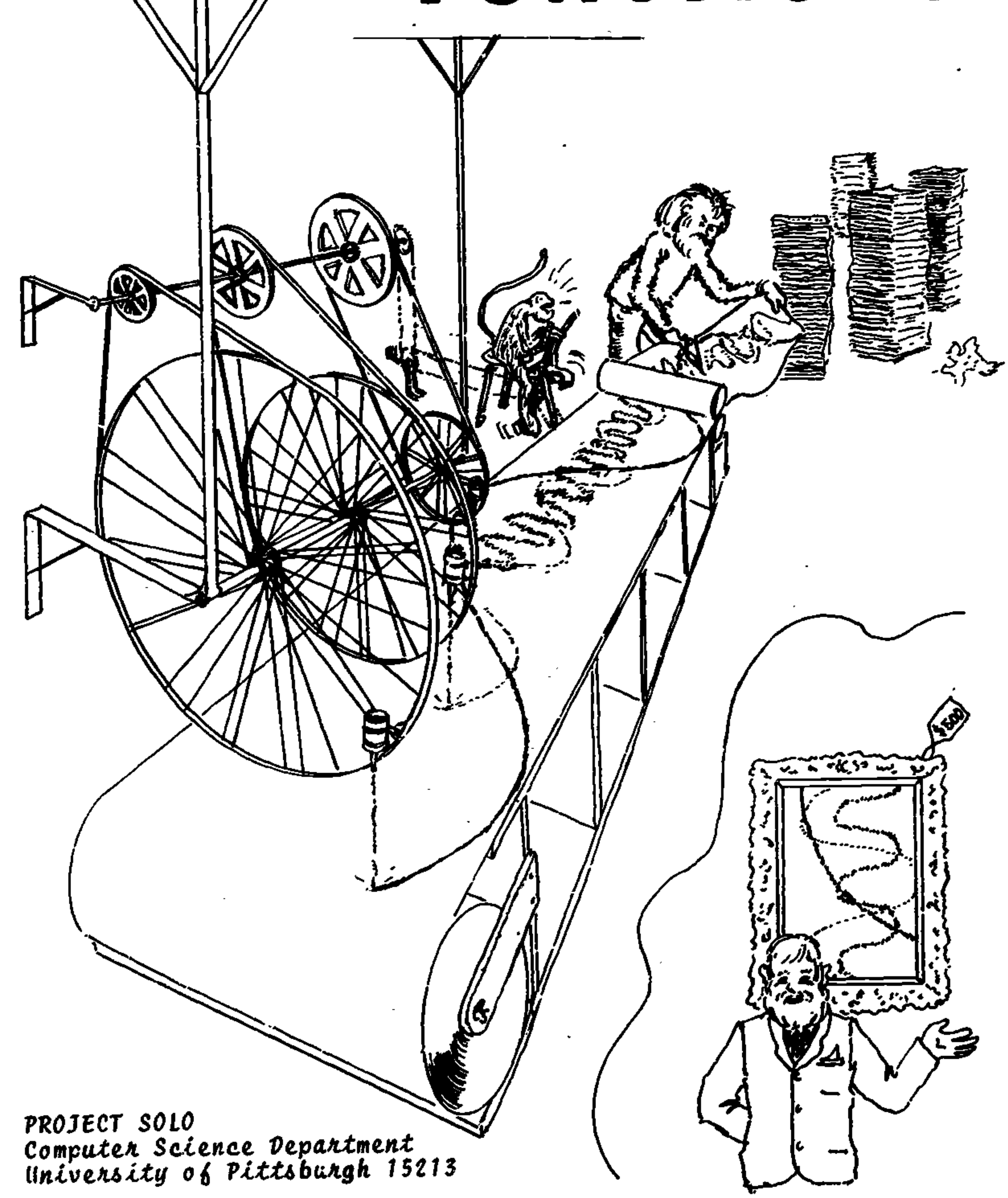
We are looking into the possibility of exchanging newsletters and programs with other high school computing efforts. The two newsletters we are aware of right now come from the Southern Minnesota School District Project at Mankato, Minneapolis, and from Brooklyn Polytech in New York (The Huntington Project). This later group is specializing in simulation (type II) programs.

EM009391

*Supported in part by NSF grant GJ 1077

ERIC

CIRCULAR FUNCTIONS



PROJECT SOLO
Computer Science Department
University of Pittsburgh 15213

CIRCULAR FUNCTIONS

Consider an angle A with its initial side placed along the X-axis, with its vertex at the origin P:(0,0), and with a terminal side that passes through the point P:(X,Y). Let us call the distance from (0,0) to (X,Y) R, so that $R^2 = X^2 + Y^2$. Then the six circular functions of the angle A are defined as:

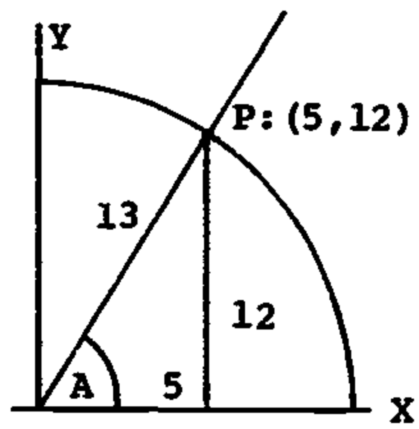
Sine of the angle = y/r	Contangent of the angle = x/y
Cosine of the angle = x/r	Secant of the angle = r/x
Tangent of the angle = y/x	Cosecant of the angle = r/y

Sample problem:

Find the circular functions of an angle (A) whose terminal side passes through the point P: (5,12).

Solution:* x = 5 and y = 12
therefore r = SQRT (5*5 + 12*12) = SQRT(169) = 13
thus sin A = 12/13 = .9231

cos A = 5/13 = .3846
tan A = 12/5 = 2.400
cot A = 5/12 = .4167
sec A = 13/5 = 2.600
csc A = 13/12 = 1.0833



Using the Computer

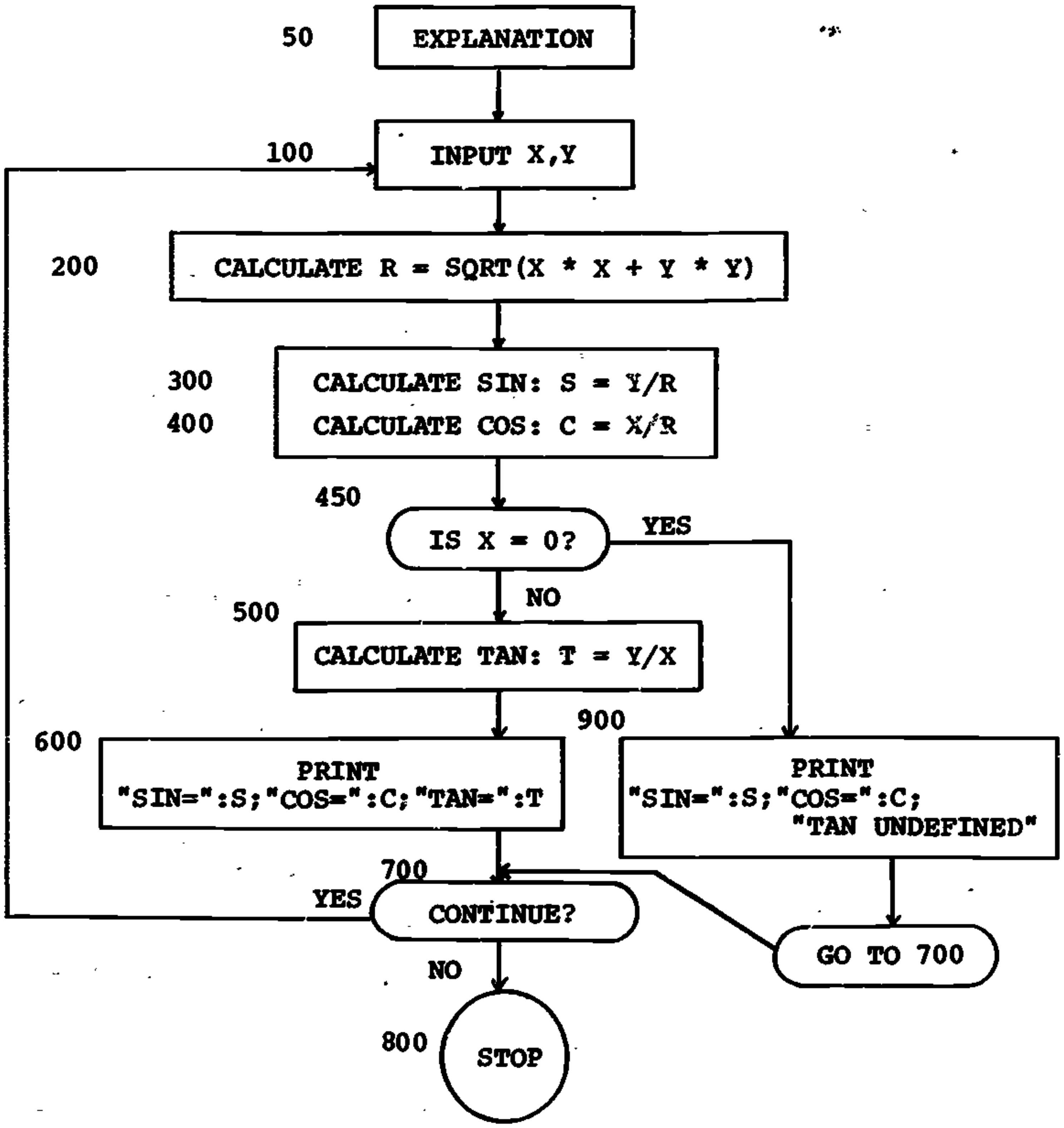
On the next page we will show the flow chart for a program that asks the user to supply values for X and Y. The program then calculates SIN A, COS A, and TAN A.

Notice that care is taken to avoid dividing by zero. (For which circular functions can this happen???)

More advanced programming problems are suggested on pages 5 and 6.

*NOTE: When using the computer you should type 5 as 5. or 5.0 and 12 as 12. or 12.0 etc.

Flowchart for a Program to Calculate Sin, Cos, and Tan



Here is an NBS program based on the flow chart on page 3:

```
50 PR. "THIS PROGRAM CALCULATES THE SIN, COS, AND TAN FUNCTIONS"
60 PR. "OF AN ANGLE A DEFINED BY THE RATIOS Y/R, X/R, AND Y/X "
70 PR. "RESPECTIVELY"
95 PR.
100 PR. "TYPE IN THE COORDINATES OF YOUR POINT IN X,Y ORDER, USING"
110 PR. "DECIMAL FORM"
120 INPUT X,Y
200 R=SQRT(X*X+Y*Y)
300 S=Y/R
400 C=X/R
450 IF X=0 GOTO 900
500 T=Y/X
600 PR. "SIN(A)=":S;"COS(A)=":C;"TAN(A)=":T
700 PR. "DO YOU WISH TO ENTER ANOTHER POINT";
710 INPUT R$
720 Y$="YES,SURE,OK,O.K.,CERTAINLY,YUP,OF COURSE,RIGHT,YEAH,Y,"
730 IF IEQIV(R$,Y$,1) GOTO 100
800 STOP
900 PR. "SIN(A)=":S;"COS(A)=":C;"TAN UNDEFINED"
910 GOTO 700
>RUN
THIS PROGRAM CALCULATES THE SIN, COS, AND TAN FUNCTIONS
OF AN ANGLE A DEFINED BY THE RATIOS Y/R, X/R, AND Y/X
RESPECTIVELY.

TYPE IN THE COORDINATES OF YOUR POINT IN X,Y ORDER, USING
DECIMAL FORM.

?5.0, 12.0
SIN(A)=.9231        COS(A)=.3846      TAN(A)=2.4
DO YOU WISH TO ENTER ANOTHER POINT?  SURE
TYPE IN ...........ETC.
```

<u>NOTE</u>: You must type in two decimal numbers. If you don't know a number (say the square root of 3), you can use @NBS.

<u>EXAMPLE</u>: Suppose you want to find the circular functions for P: ($\sqrt{3}$, $\sqrt{2}$).

WRONG!!

```
>RUN
......................
TYPE IN THE COORDINATES OF YOUR POINT
?SQRT(3), SQRT(2)
```

CORRECT

```
>RUN
THIS PROGRAM ETC.
......................
TYPE IN THE COORDINATES OF YOUR POINT IN X,Y ORDER,.....
?@NBS
>PR. SQRT(3)
1.732
>PR. SQRT(2)
1.414
>EXIT
?1.732, 1.414
.................ETC.................
```

Computer Problems

1. Use the program on Page 4 for P:(1,1), P:(0,1), P:(1,0), P:(-3,3), P:(-2,3), P:(-3,-3), P:(3,-3) [IMPT: Type 1 as 1.0 etc.]

2. Modify the program given on Page 4 so that all six circular functions are calculated.

   a. Use your computer program to find the circular functions of the following:

      i. An angle A whose terminal side passes through P:(2*SQRT(2), SQRT(7)) NOTE: You must use @NBS to find 2*SQRT(2) etc.

      ii. An angle A whose terminal side passes through P:(-2.386, 7.590)

      iii. An angle A whose terminal side passes through P:(0,2.7)

      iv. An angle A whose terminal side passes through P:(2.7,0)

   b. Run your program again, supplying data on several points in all four quadrants. Can you make a general rule about the signs of the circular functions in each quadrant?

   c. Investigate angles whose terminal sides are coincident with the coordinate axes; that is, circular functions of 0, 90, 180, 360°. Which functions are undefined?

Advanced Problems:

3. Write a program that automatically generates the six circular functions

   FOR X = 1.0, .9, .8, .7, .6, .5, .4, .3, .2, .1, 0.
   -.1, -.2, -.3, -.4, -.5, -.6, -.7, -.8, -.9, -1.0

   WITH R FIXED AT R = 1 (UNIT CIRCLE).

   Here is part of a program to do this with some output:

```
100 A$="2(SD.DDB) 2(SD.DDDB) 3(SDD.DDDB) SDD.DDDB/"
110 B$="2(SD.DDB) 2(SD.DDDB) ' UNDEF  ' SDD.DDDB ' UNDEF  ' SDD.DDDB/"
120 C$="2(SD.DDB) 2(SD.DDDB) SDD.DDDB ' UNDEF  ' SDD.DDDB' UNDEF  '/"
130 R=1
140 PRINT " X     Y     SIN    COS    TAN    COT    SEC    CSC"
150 FOR X=1 BY -.1 WHILE X>-1.1
160 IF ABS(X)>=1 GOTO 280
170 Y=ABS(SQRT(R*R-X*X))
180 IF ABS(X)<.005 GOTO 250
190 IF ABS(Y)<.005 GOTO 280
200 PRINT IN FORM A$:X:Y:Y/R:X/R:Y/X:X/Y:R/X:R/Y
210 NEXT X
230 END
```

3. (Continued)

.....ETC.
>RUN

X	Y	SIN	CØS	TAN	CØT	SEC	CSC
+1.00	+0.00	+0.000	+1.000	+00.000	UNDEF	+01.000	UNDEF
+0.90	+0.44	+0.436	+0.900	+00.484	+02.065	+01.111	+02.294
+0.80	+0.60	+0.600	+0.800	+00.750	+01.333	+01.250	+01.667
+0.70	+0.71	+0.714	+0.700	+01.020	+00.980	+01.429	+01.400
+0.60	+0.80	+0.800	+0.600	+01.333	+00.750	+01.667	+01.250
+0.50	+0.87	+0.866	+0.500	+01.732	+00.577	+02.000	+01.155
+0.40	+0.92	+0.917	+0.400	+02.291	+00.436	+02.500	+01.091
+0.30	+0.95	+0.954	+0.300	+03.180	+00.314	+03.333	+01.048
+0.20	+0.98	+0.980	+0.200	+04.899	+00.204	+05.000	+01.021
+0.10	+0.99	+0.995	+0.100	+09.950	+00.101	+10.000	+01.005
+0.00	+1.00	+1.000	+0.000	UNDEF	+00.000	UNDEF	+01.000
-0.10	+0.99	+0.995	-0.100	-09.950	-00.101	-10.000	+01.005

.....ETC.

4. <u>Man Against Machine:</u>

Try to make the computer blow its mind by calculating TAN(A), with R = 1, and X approaching zero.

ONE APPROACH

```
10 LET X=1
20 IF X<.0000000000001 GØTØ 60
30 PRINT "X=":X:TAB(35):"TAN(X)=":SQRT(1-X*X)/X
40 LET X=X/2
50 GØTØ 20
60 END
```

5. The picture on the cover shows that the paint dripping from the paint dispensers on the three wheels (rotated by monkey power) traces out "graphs" of the circular functions on the canvas (moved by mad artist power).

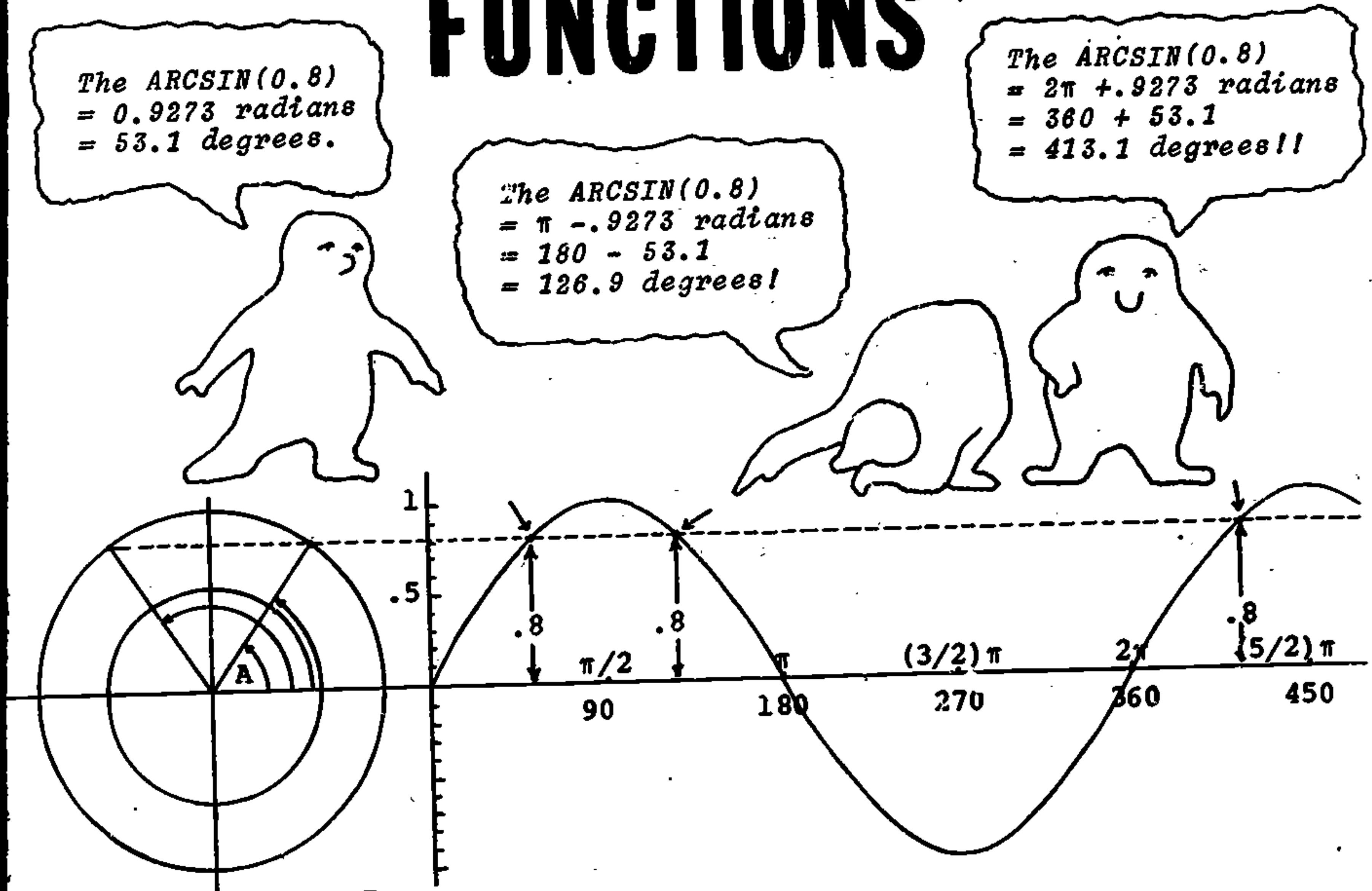
Write a program that:

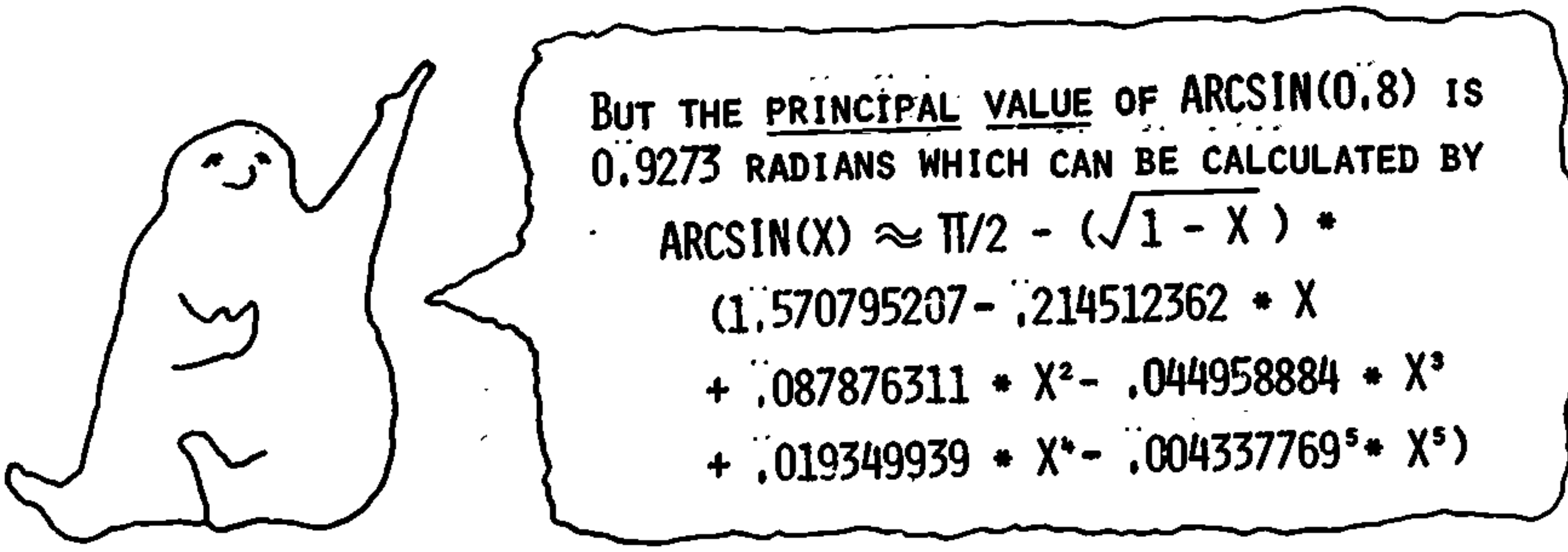
(a) Graphs the SIN or COS function.
OR (b) Graphs the SIN and COS function on the same axis, using two different values of R.
OR (c) Graphs SIN and COS functions with different values of R and different starting points, possibly making "computer generated" art as your goal.

<u>Suggestions:</u>

1. Ask your teacher for the module called "Graphing by Computer."
2. Could you make your graph print symbols like "2", "3", etc. to make a "number" painting for later water coloring, where 1 = Red, 2 = Blue, etc.

INVERSE CIRCULAR FUNCTIONS





PROJECT SOLO
Computer Science Department
University of Pittsburgh 15213

FINDING THE INVERSE CIRCULAR FUNCTIONS

In the module on trigonometric functions we saw that polynomial approximations are used by computers to calculate the circular (or trigonometric) functions when the independent variable is the angle.----- The simple NBS program:

```
>10 LET A=.9273
>20 PRINT "A=":A;"SIN(A)=":SIN(A)
>RUN
```

will result in the output:

```
A=.9273 SIN(A)=.800002869
```

In order that the computer successfully execute this program it must use one of the Hastings approximations for finding SIN(.9273). (See Problem 3 in the module "TRIG. FUNCTIONS AND TCHEBYCHEV APPROXIMATIONS".)

THE INVERSE PROBLEM

Now suppose we ask the computer to

```
>PR. ARCSIN(.8)
```

we will obtain:

```
0.927295218
```

We have asked the computer to print:

"the angle which has a sine of 0.8"

or to be exact:

"Print the angle A (in radians) such that SIN(A)=0.8, and A lies in the interval $-\pi/2 \leq A \leq \pi/2$

The cover of this module shows why we have to agree on an interval for A. The inverse function ARCSIN is not single valued, and so we must eliminate confusion by using the (unspoken) assumption that the principal value of A will be calculated. For the SINE function, the principal value is found in either quadrant I or quadrant IV.

HASTINGS SHEET 35

The formula for approximating ARCSIN(X) shown on the front cover is Hastings' Sheet 37, a rather accurate formula. It assumes that X is limited to the range

$$0 \le X \le 1$$

Here is an NBS program to use a simpler but less accurate formula (Hastings' Sheet 35) to calculate the ARCSIN function. The program also prints the more accurate NBS ARCSIN function for comparison. The Sheet 35 formula is:

$$\text{ARCSIN}(X) \doteq \pi/2 - (\sqrt{1-X}) * (1.5707288 - .2121144*X + .0742610*X*X - .0187293*X*X*X)$$

Notice that polynomial is evaluated using nested multiplication in line 560 in order to reduce the number of multiplications. Why is this important?

```
PRØGRAM TØ CØMPARE NBS LIBRARY ARCSIN FUNCTIØN
WITH HASTINGS 35 FUNCTIØN

ARGUMENT            NBS LIBRARY         HASTINGS 35
--------------------------------------------------
 0                   0                   6.620000204E-05
 0.1                 0.100167421         0.100107027
 0.2                 0.201357921         0.201313803
 0.3                 0.304692654         0.304700575
 0.4                 0.411516846         0.411559864
 0.5                 0.523598776         0.523643617
 0.6                 0.643501109         0.643521213
 0.7                 0.775397497         0.775385477
 0.8                 0.927295218         0.927265853
 0.9                 1.119769515         1.119751877
 1                   1.570791655         1.570789605

>LISTNH

500 PR."PRØGRAM TØ CØMPARE NBS LIBRARY ARCSIN FUNCTIØN "
501 PR."WITH HASTINGS 35 FUNCTIØN" PR. PR.
510 LET A0=1.5707288 LET A1=-.2121144
520 LET A2=.0742610 LET A3=-.0187293
530 P=3.14159/2
540 PR."ARGUMENT":TAB(20):"NBS LIBRARY":TAB(40):"HASTINGS 35"
545 PR."--------------------------------------------------"
550 FØR X=0 TØ 1 STEP 0.1
560 D=((A3*X+A2)*X+A1)*X+A0
570 S=P-SQRT(1-X)*D
580 PR.X:TAB(20):ARCSIN(X):TAB(40):S
590 NEXT X
600 END
```

Problem 1

Improve the above program so that X may lie in the interval $-1 \le X \le 1$. Test your program by running it with line 550 changed to:

```
550 FOR X=-1 TO 1 STEP 0.1
```

[HINT: RECALL THAT SIN(-X)=-SIN(X)]

Problem 2

Find out what NBS does when you type

```
>PR. ARCSIN(2)
```

Can you make your program do the same for the Hastings 35 approximation? Or do you have to?

Problem 3 (OPTIONAL)

Change your program to use the Hastings Sheet 36 formula:

```
ARCSIN(X)≐π/2-(√1-X)*(1.57078786-0.21412453*X+0.08466649*X*X
        -0.03575663*X*X*X+0.00864884*X*X*X*X)
```

Use π=3.14159265

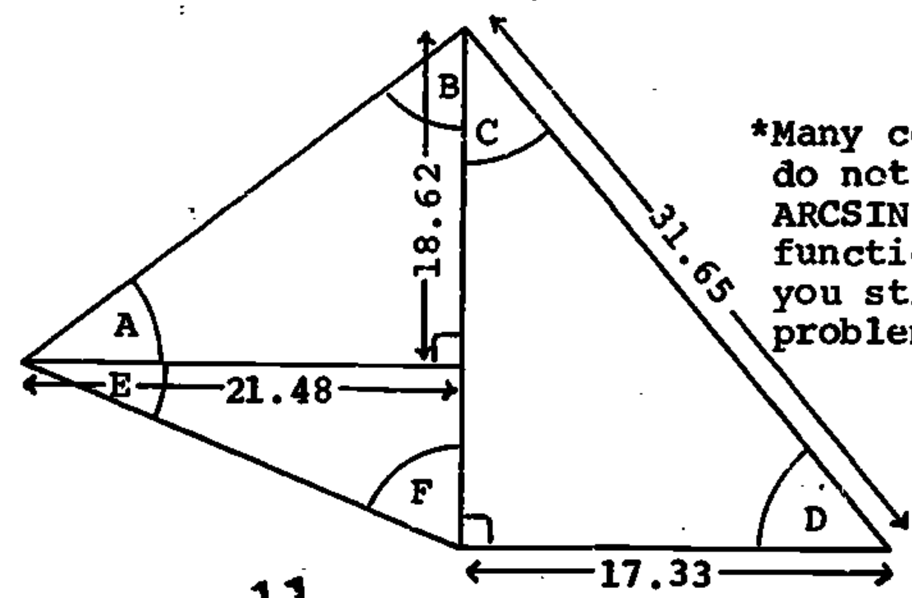
Problem 4 (OPTIONAL)

Run the program for Problem 3 again, using Hastings' 37 (on cover) and the Double Precision feature of NBS. (See NBS Primer, Section 7, "EXTENDED DATA TYPES". Use π=3.141592653589793238)

Problem 5

In practice, the library functions of NBS are used for calculation. Practice using the ARCSIN(X), the ARCCOS(X), and the ATAN(X) functions*in NBS to find the angles A, B, C, D, E, F in the figure below. (Use direct mode if you wish.)

A=
B=
C=
D=
E=
F=



*Many computers do not have ARCSIN or ARCCOS functions. Can you still do this problem?

Trigonometric Functions & Tchebychev Approximations

(OR--CAN A HUMBLE COMPUTER THAT ONLY KNOWS HOW TO DO ARITHMETIC CALCULATE TRANSCENDENTAL FUNCTIONS?)

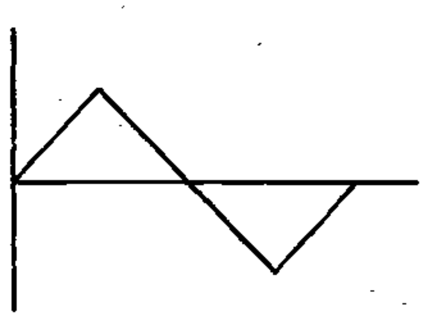
Problem 1. Below you see three representations of the "sine" function. Which is the right one?

- ☐ 1. (a)
- ☐ 2. (b)
- ☐ 3. (c)
- ☐ 4. All of the above.
- ☐ 5. None of the above.

(a) The SINE function:

Y	Y/R
0.0	0.000
0.1	0.100
0.2	0.200
0.3	0.300
0.4	0.400

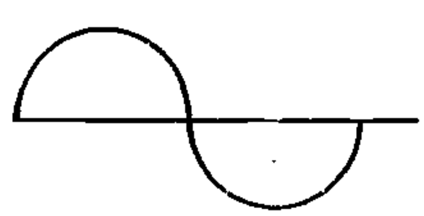
etc.



(b) The SINE function:

X	$\pm\sqrt{(R^2-X^2)}/R$
1.0	0.000
0.9	0.436
0.8	0.600
0.7	0.714
0.6	0.800

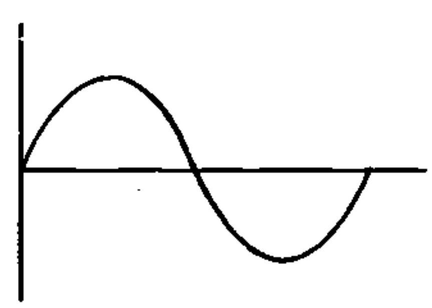
etc.



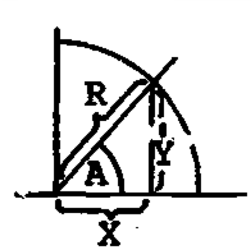
(c) The SINE function:

A	SIN(A)
0.0	0.000
0.1	0.099
0.2	0.198
0.3	0.295
0.4	0.389

etc.



Note: We are using the letters X, Y, A, and R to represent the same quantities as in the module "Circular Functions".



<u>Answer to Problem 1:</u> All of the representations are correct.

Here are three NBS programs which you can try to convince yourself that you can get three different-looking "graphs" of the sine function. The shape of these graphs depends on which variable (X, Y, or A) you decide to increment in equal steps.

After looking over these programs, you may get the feeling that something has been 'put over' on you. Question: Has a little mathematical "hanky-panky" crept into these demonstration programs???? Which one??? (For the answer, read on!!!)

<u>Demonstration Program (a)</u>

```
                              +2
                              + 2
                              +    2
                              +       2
                              +          2
                              +             2
                              +                2
                              +                   2
                              +                      2
                              +                         2
                              +                             2
                              +                         2
                              +                      2
                              +                   2
                              +                2
                              +             2
                              +          2
                              +       2
                              +    2
                              + 2
                             2+
                           2  +
                        2     +
                     2        +
                  2           +
               2              +
            2                 +
         2                    +
      2                       +
   2                          +
2                             +
   2                          +
      2                       +
         2                    +
            2                 +
               2              +
                  2           +
                     2        +
                        2     +
                           2  +
                             2+
```

```
200 FØR Y=0 STEP .1 UNTIL Y>.95
210 PRINT TAB(30):"+":TAB(30+30*Y):"2"
220 NEXT Y
230 FØR Y=1 STEP -.1 UNTIL Y<.05
240 PRINT TAB(30):"+":TAB(30+30*Y):"2"
250 NEXT Y
260 FØR Y=0 STEP -.1 UNTIL Y<-.95
270 PRINT TAB(30+30*Y):"2":TAB(30):"+"
280 NEXT Y
290 FØR Y=-1 STEP .1 UNTIL Y>.05
295 PRINT TAB(30+30*Y):"2":TAB(30):"+"
296 NEXT Y
299 END
```

Demonstration Program (b)

```
                              +1
                              +            1
                              +                 1
                              +                    1
                              +                       1
                              +                        1
                              +                          1
                              +                           1
                              +                            1
                              +                            1
                              +                             1
                              +                            1
                              +                            1
                              +                           1
                              +                          1
                              +                        1
                              +                       1
                              +                    1
                              +                 1
                              +            1
                              1+
                 1            +
            1                 +
         1                    +
      1                       +
     1                        +
   1                          +
  1                           +
 1                            +
 1                            +
1                             +
 1                            +
 1                            +
  1                           +
   1                          +
     1                        +
      1                       +
         1                    +
            1                 +
                 1            +
```

```
100 FØR X=1 STEP -.1 UNTIL X<-.95
110 LET S=SQRT(1-X*X)
120 PRINT TAB(30);"+";TAB(30+30*S);"1"
130 NEXT X
140 FØR X=-1 STEP .1 UNTIL X>.95
150 LET S=-SQRT(1-X*X)
160 PRINT TAB(30+30*S);"1";TAB(30);"+"
170 NEXT X
180 END
```

Demonstration Program (c)

```
300 FØR A=0 STEP 6.283/40 UNTIL A>6.283
310 LET S=SIN(A)
320 IF S>0 THEN PR. TAB(30):"+":TAB(30+30*S):"3" ELSE PR.
TAB(30+30*S):"3":TAB(30):"+"
330 NEXT A
340 END
```

```
                             3+
                              +   3
                              +        3
                              +            3
                              +                 3
                              +                     3
                              +                        3
                              +                          3
                                                           3
                                                            3
                                                            3
                                                            3
                                                           3
                                                         3
                                                       3
                                                    3
                              +                 3
                              +            3
                              +        3
                              +   3
                              +3
                         3    +
                    3         +
                3             +
            3                 +
        3                     +
     3                        +
   3                          +
 3                            +
3                             +
3                             +
3                             +
 3                            +
   3                          +
     3                        +
        3                     +
            3                 +
                3             +
                    3         +
                         3    +
                             3+
```

If you haven't spotted the hanky-panky by now, check line 310 above. Assuming computers can only do arithmetic (which is true), an expression like 30 + 30 * S in line 320 makes sense, but it isn't clear that arithmetic is being used in line 310. Where is the arithmetic being done? Answer: SECRETLY! The computer actually goes into a "library" sub-routine at line 310, which tells it what arithmetic operation must be done on the angle A (expressed in radians) in order to calculate SIN(A).

The Secret Unveiled

The library functions in digital computers used to evaluate complicated functions almost always employ expertly designed polynomials as approximations to these functions. These approximations can be made quite accurate, using techniques originated by the Russian mathematician Tchebychev (also spelled Chebyshev), and put into useful form by Hastings.[1]

Here is one of the approximations given by Hastings: (Hastings Sheet 14)

Function: $\sin\left(\frac{\pi}{2} X\right)$

Range: $-1 \le X \le 1$

Approximation: $\sin\left(\frac{\pi}{2} X\right) \approx C_1X + C_3X^3 + C_5X^5$

where $C_1 = 1.5706268$, $C_3 = -.6432292$, and $C_5 = .0727102$.

Before using this approximation, it will be useful to make two changes. First, we will change the range of the independent variable by the substitution:

$Y = (\pi/2) * X$ so that when: $-1 \le X \le 1$

we will have: $-\pi/2 \le Y \le \pi/2$

Note: Both Y and X are angles, measured in radians. Don't confuse these variables with the Y and X coordinates used on the cover. The variable Y in the rest of this discussion is identical to the variable A shown on the cover.

After making this substitution, the Hastings Sheet 14 approximation becomes:

$$\sin(Y) \approx K_1Y + K_3Y^3 + K_5Y^5$$

where $K_1 = (2/\pi) * C_1$; $K_3 = (2^3/\pi^3) * C_3$; $K_5 = (2^5/\pi^5) * C_5$.

[1] A book your library ought to have is Hastings, Cecil, Approximations for Digital Computers, Princeton University Press, 1955.

The second change we will make is to rearrange the right side of the preceding formula into "nested multiplication" form:

$$\sin(Y) \approx (((K_5 * Y * Y + K_3) * Y) * Y + K_1) * Y$$

The advantage of this form is that it only takes five multiplications, whereas the original polynomial takes nine.

Here is an NBS program which uses this form to calculate sin(Y), and compares it with the NBS library routine (which uses a higher degree approximation--see problem 3).

```
PRØGRAM TØ CØMPARE NBS LIBRARY SINE FUNCTIØN WITH
HASTINGS 14.


RADIANS             NBS LIBRARY         HASTINGS 14
-----------------------------------------------------
 -1                 -0.841470985        -0.841534239
 -0.9               -0.78332691         -0.783406887
 -0.8               -0.717356091        -0.717433028
 -0.7               -0.644217687        -0.644277689
 -0.6               -0.564642473        -0.564678888
 -0.5               -0.479425539        -0.47943851
 -0.4               -0.389418342        -0.389413182
 -0.3               -0.295520207        -0.295505151
 -0.2               -0.198669331        -0.19865316
 -0.1               -0.099833417        -0.099823323
  0                  0                   0
  0.1                0.099833417         0.099823323
  0.2                0.198669331         0.19865316
  0.3                0.295520207         0.295505151
  0.4                0.389418342         0.389413182
  0.5                0.479425539         0.479438509
  0.6                0.564642473         0.564678888
  0.7                0.644217687         0.644277689
  0.8                0.717356091         0.717433028
  0.9                0.78332691          0.783406887
  1                  0.841470985         0.841534239

>LISTNH

290 PR."PRØGRAM TØ CØMPARE NBS LIBRARY SINE FUNCTIØN WITH"
291 PR."HASTINGS 14." PR. PR.
300 LET P=3.14159265
310 LET P2=P*P LET P3=P*P2
320 LET C1=(1.5706268*2)/P
330 LET C3=(-.6432292*8)/P3
340 LET C5=(.0727102*32)/(P2*P3)
350 PR. "RADIANS":TAB(20):"NBS LIBRARY":TAB(40):"HASTINGS 14"
355 PR."-----------------------------------------------------"
360 FØR Y=-1 TØ 1 STEP 0.1
365 IF ABS(Y)<.0001 LET Y=0
370 S1=(((C5*Y*Y+C3)*Y)*Y+C1)*Y
380 PR. " ":Y:TAB(20):SIN(Y):TAB(40):S1
390 NEXT Y
400 END
```

Problem 2. Change the FOR loop in the above program to cover the range for Y = 1 to 3 radians, and see what happens. Also try the range for Y = -1 to -3. If something "blows up," can you modify your program so that *any* value of Y can be used in the Hastings' approximation? (Hint: recall $\sin(Y) = \sin(\pi - Y) = \sin(2\pi + Y) = \sin(3\pi - Y) = $ etc.)

Problem 3. Here is the Hastings Sheet 15 approximation which ought to come a lot closer to the values produced by the NBS library function. Modify your program to try it out, again comparing it with the NBS library function.

$$\sin((\pi/2) * X) \approx C_1 * X + C_3 * X^3 + C_5 * X^5 + C_7 * X^7$$

where

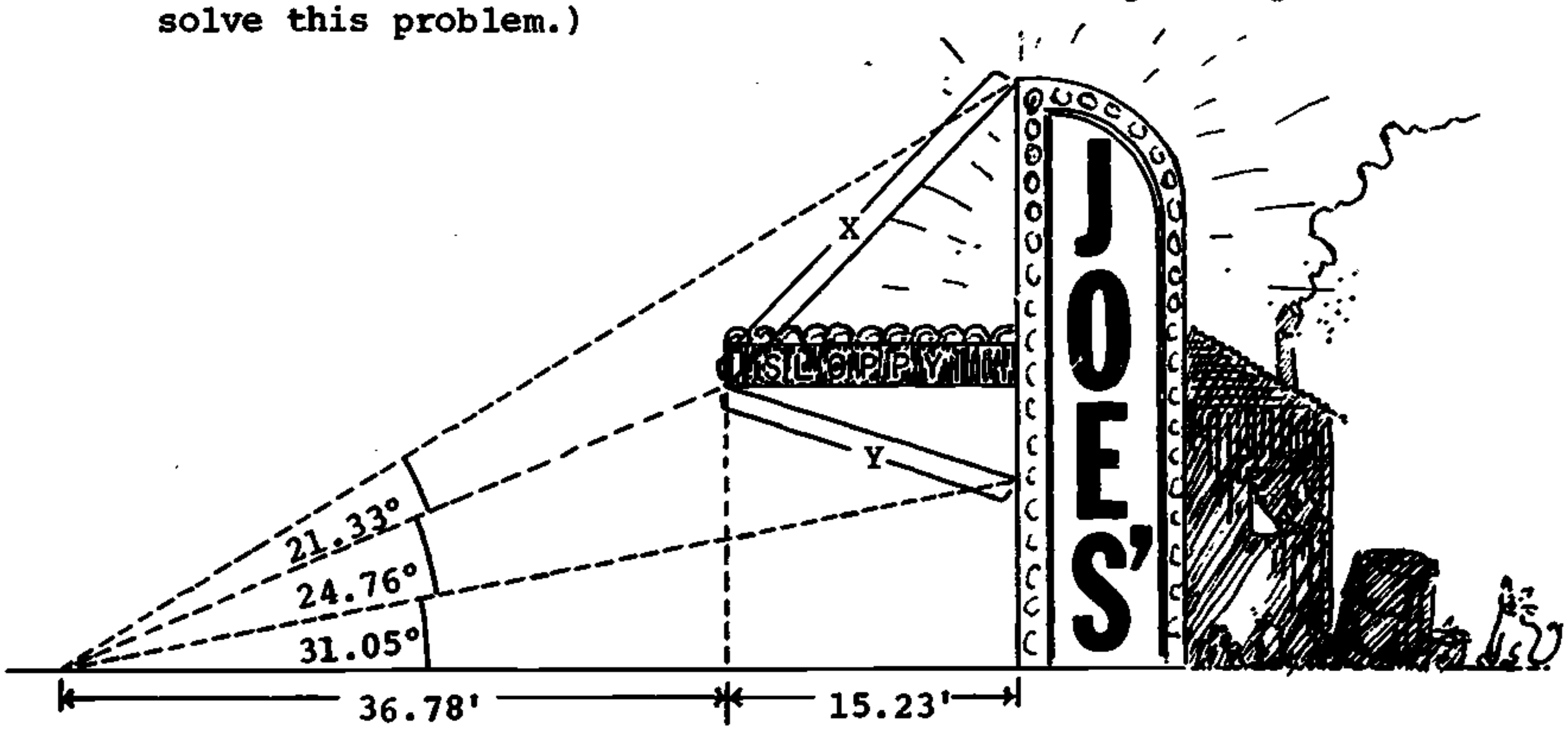
$C_1 = 1.570794852$ $\qquad C_3 = -0.645920978$

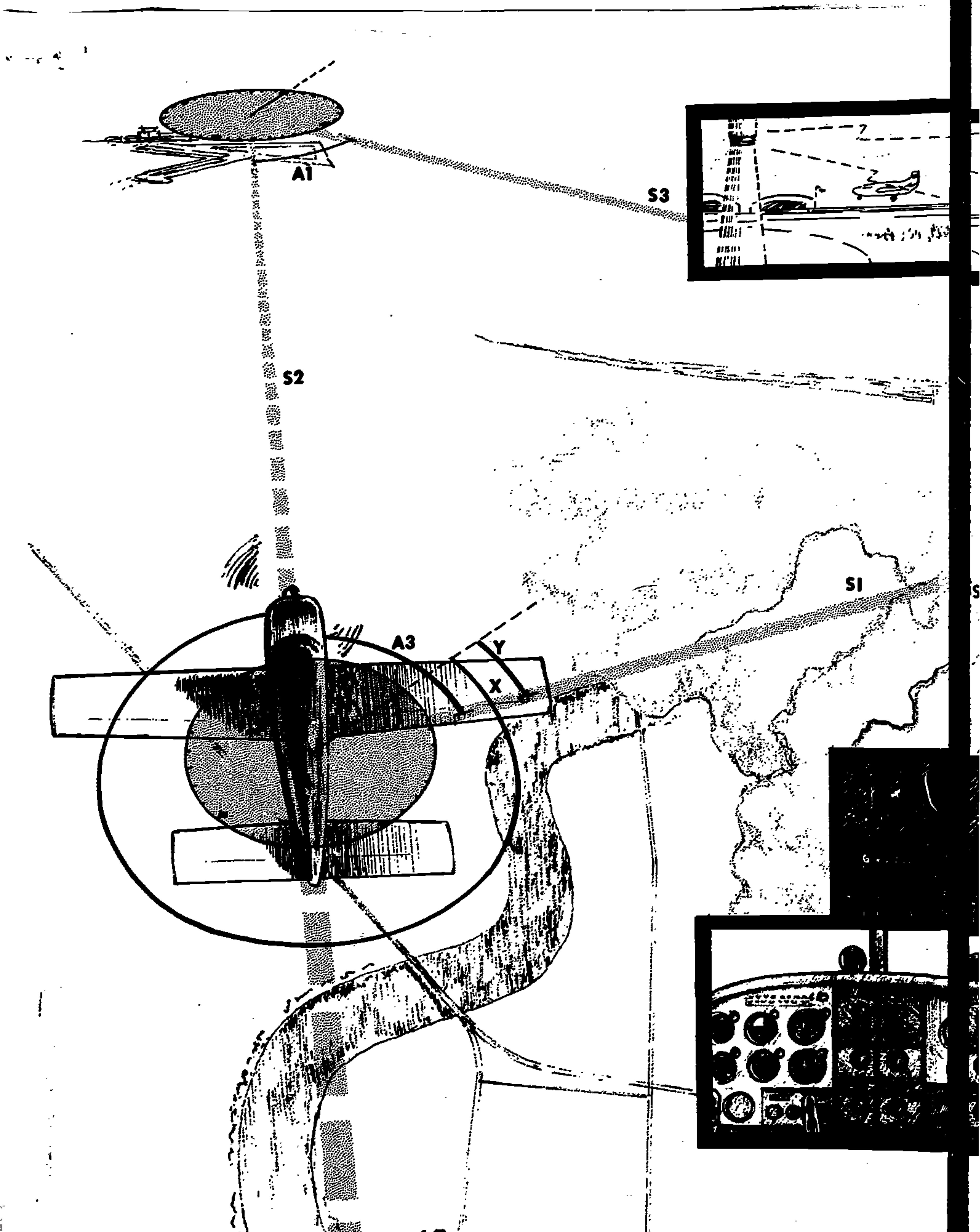
$C_5 = 0.079487663$ $\qquad C_7 = -0.004362476$

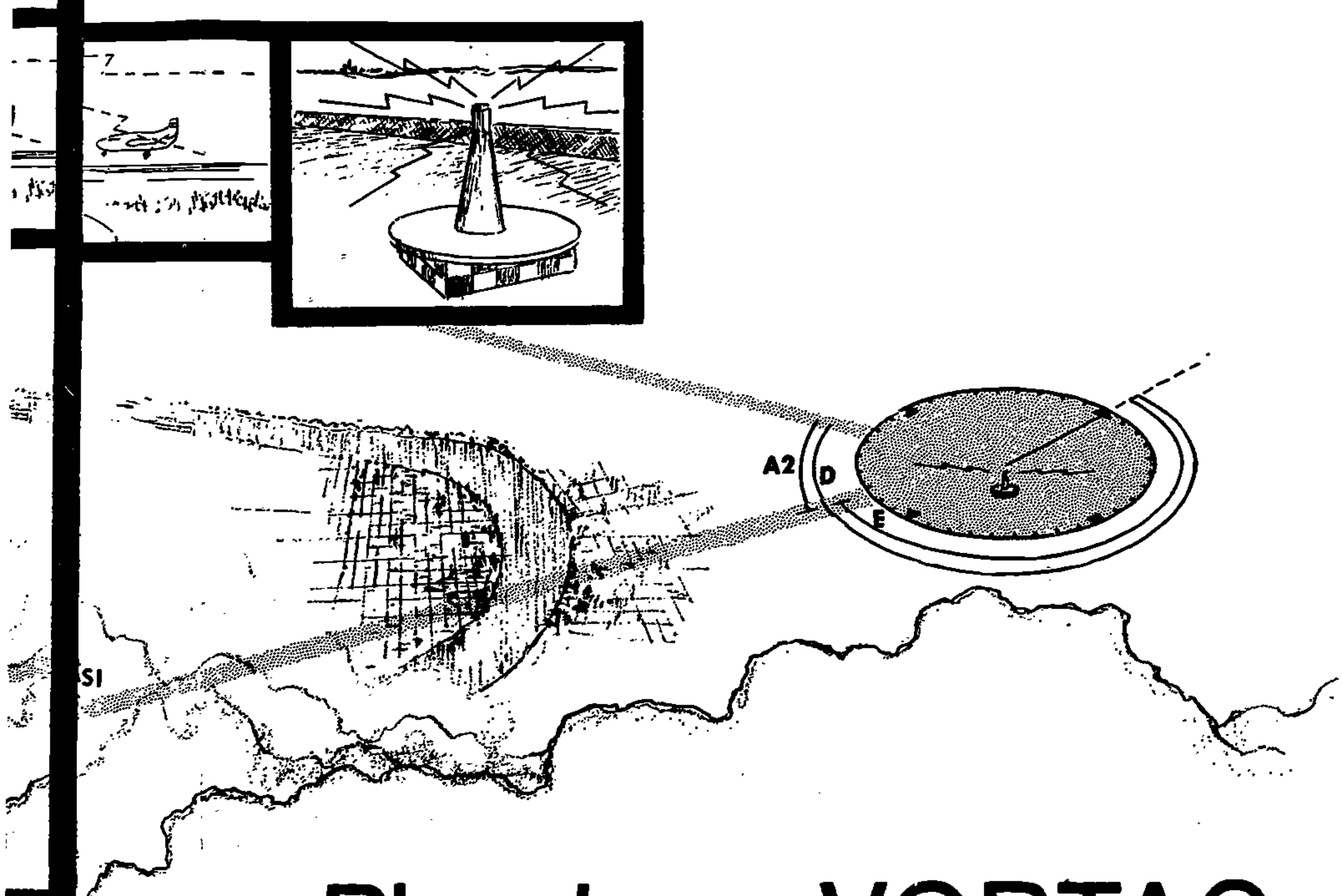
and $\qquad -1 \leq X \leq 1$

Problem 4. (Optional) If you find the idea of super-precision of interest, ask your teacher to see the Com-Share XTRAN Library Manual, and explore the use of double precision arithmetic and double precision library functions. One of the best handbooks to use as a standard for "correct" values is "The Handbook of Mathematical Functions, Applied Math Series No. 55" available from the Superintendent of Documents, U.S. Government Printing Office, Washington, D. C. 20402 ($6.50). A paperback version published by Dover Co. may be available at book stores for about $5.00.

Problem 5. (Easy) Use the NBS library functions TAN and SQRT to find the lengths of the guy-wires X and Y shown below. (Use of direct mode NBS as a desk calculator is a good way to solve this problem.)




A1
S3
S2
S1
A3
Y
X



Phantom VORTAC

● This module will guide you in preparing the master program for an on-board flight computer. The computer takes information from a VHF omni range system (VORTAC), and calculates the magnetic course and distance to a given airport for the pilot. The output of the program is such that the pilot can fly toward a "phantom" VORTAC located at any airport he selects.

● A description of this newly developed navigational system and the mathematics on which it is based are given on page 2.

● Pages 3 and 4 outline alternate methods for handling the computation, and suggest ways in which previous programs you have written might be incorporated as sub-routines.

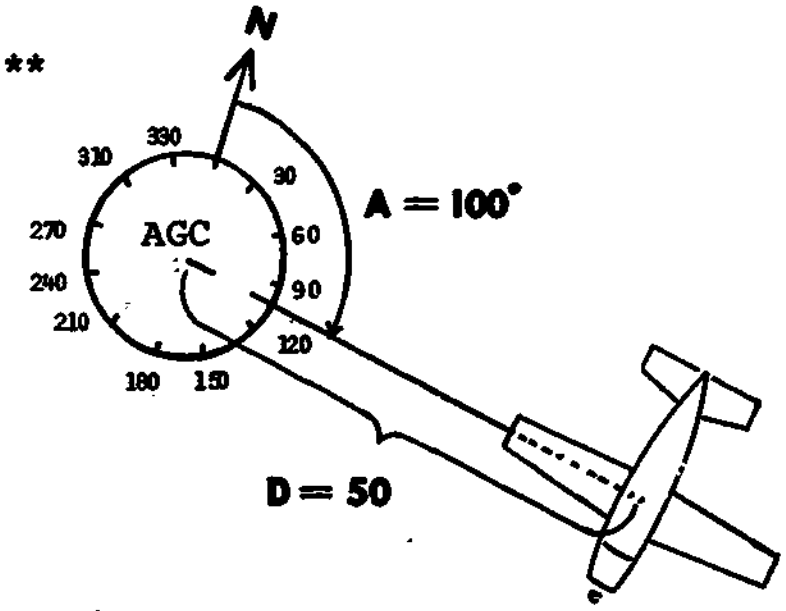
● A "real time" simulation of a flight using this system is suggested as an advanced level program.

ERIC

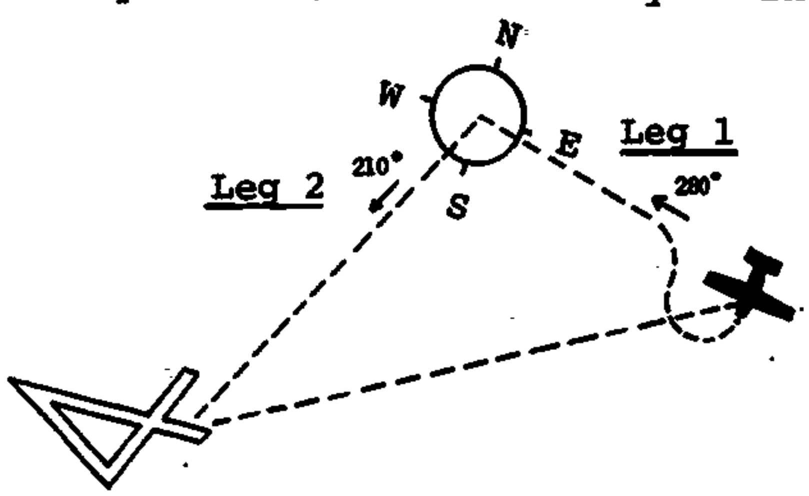
Pilots flying over the United States (and most other countries of the world) rely on radio facilities called VORTACs* for navigational information. The basic information the pilot receives in the cockpit is his position relative to the VORTAC, given in polar coordinates.

The pilot in the illustration below would describe his position (obtained from his radio instruments) as being "on the 100° radial of the Allegheny County (AGC) VOR, 50 miles out".

It is easy for this pilot to note that he can get to AGC by turning right, and flying a <u>course</u>** of 280° (Why 280°?). If he is going 200 miles per hour, it is also easy for him to estimate that he will arrive at AGC in 15 minutes (Is this exactly true? --go back over the module on vector addition if you are not sure.)



The catch to all of this is that the location of the VOR is usually different from the location of the destination airport. This difficulty can be handled by flying two legs, the first from the present position to the VOR, and the second from the VOR to the airport. This is obviously an inefficient route.



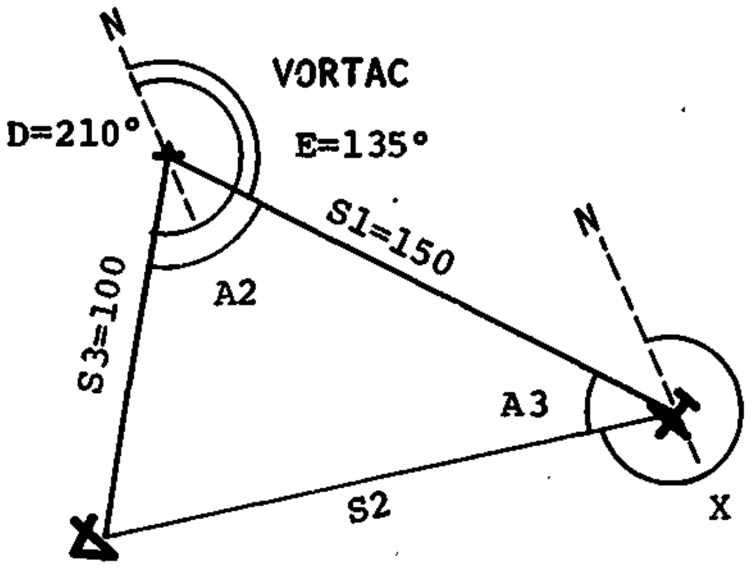
A new navigational system uses an on-board, special-purpose computer to tell the pilot what course and distance to fly in order to go <u>directly</u> from his present position to the airport. Let's first examine three ways of analyzing the mathematics involved in such a computation.

---

*<u>V</u>ery high frequency <u>O</u>mni <u>R</u>ange and <u>TACAN</u>, where TACAN is an older military system of distance measuring equipment. Most civilian pilots call these facilities VOR-DME stations.

**The angular direction of the intended flight path, measured clockwise from N.

Before writing a program for such an on-board computer, it would be useful for a programmer to solve by hand typical problems arising in this situation.* At this point, let's look at three such solutions.



Problem 1. The aircraft is 150 miles out on the 135° radial and the destination airport is 100 miles out on the 210° radial. That is, in the diagram, S1 = 150, E = 135°, S3 = 100, D = 210°.

Let A2 be the angle determined by S1 and S3. Hence, A2 = 210° - 135° = 75°. By the Law of Cosines,

$$S2 = \sqrt{100^2 + 150^2 - 2 \cdot 100 \cdot 150 \cdot \cos(75°)} = 157.3 \text{ miles.}$$

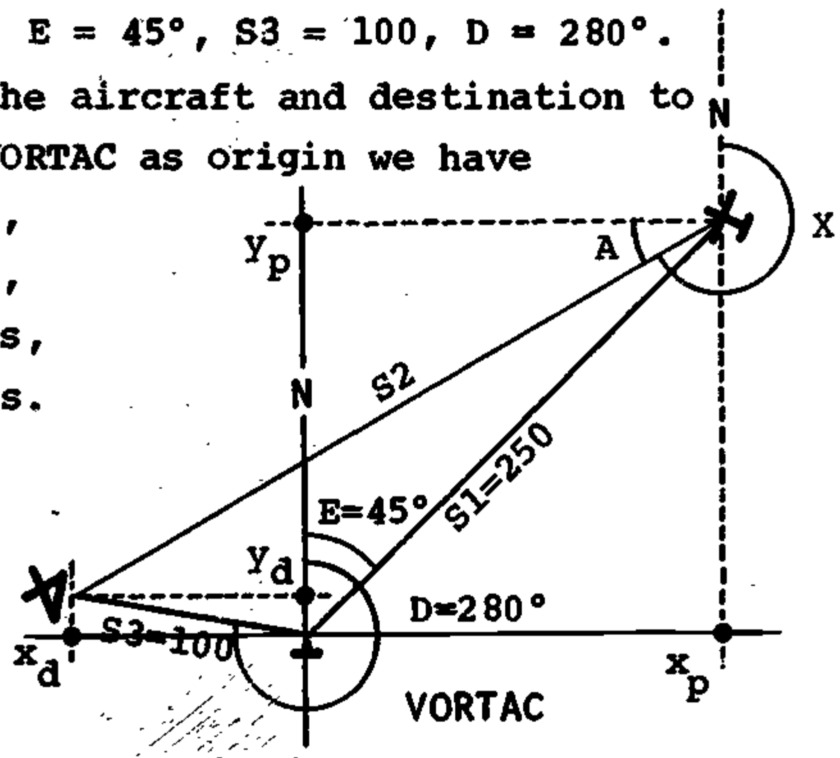
Now by the Law of Sines, $\sin A3 = (100 \cdot .966)/157.3 = .614$ and $\cos A3 = \sqrt{1 - .614^2} = .789$. From this we can get A3 = arctan(.614/.789) = 38°. (Why wasn't A3 computed directly from sin A3?) Therefore X = 135° + 180° - 38° = 277°. Output to the pilot is 157.3 miles, 277°--is this answer reasonable?

Problem 2. The airplane is 250 miles out on the 45° radial and the destination airport is 100 miles out on the 280° radial. In terms of the diagram, S1 = 250, E = 45°, S3 = 100, D = 280°.

Converting the positions of the aircraft and destination to rectangular coordinates with the VORTAC as origin we have

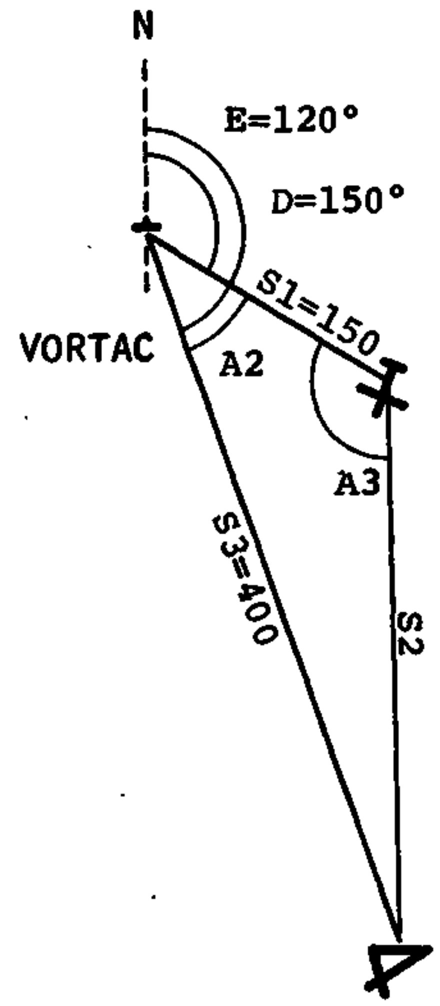
$x_p = 250 \cdot \sin 45° = 176.8$ miles,
$y_p = 250 \cdot \cos 45° = 176.8$ miles,
$x_d = 100 \cdot \sin 280° = -98.48$ miles,
$y_d = 100 \cdot \cos 280° = 17.36$ miles.

$$A = \arctan \frac{176.8 - 17.36}{176.8 - (-98.48)} = \arctan (.5792) = 30.1°$$



*This is another way of saying that computers do not remove the responsibility of analyzing a problem before solving it--quite the contrary; they demand more thought than ever. This may be, in fact, one of the most important contributions computing systems can make to learning.

Therefore X = 270° - 30.1° = 239.9°. By the Pythagorean Theorem $S2 = \sqrt{(176.8 - 17.36)^2 + (176.8 - (-98.48))^2} = 318.1$. Hence the output to the pilot is: 318.1 miles, 239.9°.



Problem 3. The aircraft is 150 miles out on the 120° radial and the destination airport is 400 miles out on the 150° radial.** In the diagram, E = 120°, S1 = 150 miles, D = 150°, S3 = 400 miles. Using the same reasoning as in Problem 1 yields:

$A2 = 30°$

$S2 = \sqrt{400^2 + 150^2 - 2 \cdot 150 \cdot 400 \cdot \cos 30°}$

$= 280.3$ miles

$\sin A3 = 400 \cdot (.5236/280.3) = .7135$

$\cos A3 = \sqrt{1 - .7135^2} = .7007$

$A3 = \arctan(.7135/.7007) = 45.52°$

$X = 120° + 180° - 45.52° = 254.48°$

Output to the pilot is: 280.3 miles, 254.48°--does this answer look reasonable? What went wrong?

Assignment. Write a program for an on-board navigational computer which will accept as input the radial and distance from a VORTAC of both an aircraft and a destination, and which will compute a course and distance to the destination. The trigonometric subroutines in the computer require arguments in radians, but pilots think in terms of degrees, so it will be necessary for you to convert degrees to radians and back (see the module on converting to radians). Other possibly useful topics are inverse trigonometric functions*, the Laws of Sines and Cosines*, and transformation of polar to rectangular coordinates*.

---

*See modules you have worked previously.

**By now you should have noticed that the word radial is used to designate the angular position of a line segment that starts at the VORTAC.

Optional Section. During most flights, the aircraft moves through the air and the air moves as well. Write an addition to your program which will:

1. Accept as additional input, specified by the 'pilot', (a) the aircraft's speed, (b) the speed of the wind, (c) the direction of the wind, (d) an elapsed time, t, and (e) a heading* for the aircraft. Any heading should be acceptable: the 'pilot' should be able to fly wherever he likes.

2. Compute the new position of the aircraft on the basis of the above information.

3. Repeat steps 1 and 2 as often as desired or necessary to reach the destination airport.

You can look at the input/output sequence of one program that accomplishes this and has been written and stored.

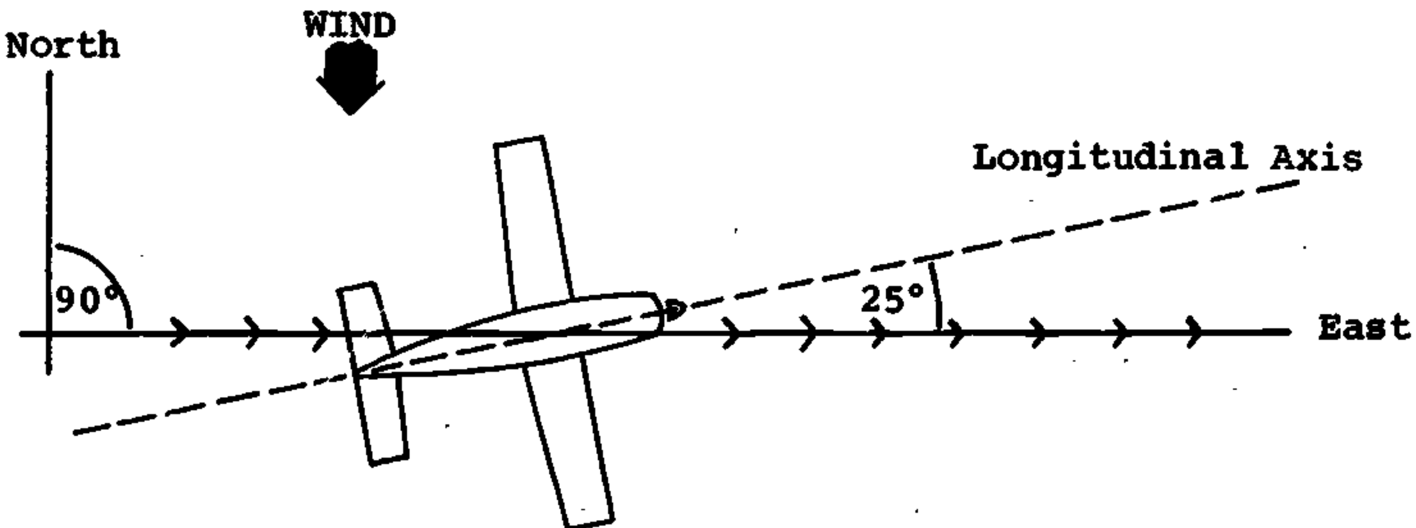
In order to gain access to this program, please logon (directions are given in an introductory module) using the User Identification Code given to you by your teacher:

In order to interact with this stored program type in the statements:

From this point on the operation of the program should be self-explanatory.

---

*Heading is defined as the angular direction of the longitudinal axis of the aircraft with respect to North. In the picture below, the pilot is flying a course of 90°, but his heading is 65°. Why?



GROSS FLOW CHART FOR SUGGESTED ASSIGNMENTS